CLAIMS

What is claimed is:

1. A method comprising:

determining a register format of a source register operated on by a source instruction in a source block of code, the register format including an input instruction format and an output block format of the source block of code, the source block of code running in a source architecture, the source register having multiple formats and being used as an input of the source instruction, the input instruction format containing format of the source register expected by the source instruction, the output block format containing format of the source register after the source block of code is executed; and

detecting an instruction format inconsistency between the source register and a target register of a target architecture during a translation phase of a binary translation that translates the source block of code into a target block of code running in the target architecture.

2. The method of claim 1 wherein detecting the instruction format inconsistency comprises:

comparing the output block format to the input instruction format if the output block format asserts an access status of the source register.

3. The method of claim 2 further comprising:

emitting a conversion code to convert the source register from the output block format to the input instruction format into the target block of code during the translation phase if the output block format is different from the input instruction format and the output block format asserts an access status of the source register.

4. The method of claim 1 further comprising:

updating an input block format and the output block format, the input block format containing format of the source register expected by the source block of code before execution.

5. The method of claim 4 wherein updating comprises:

setting the input block format and the output block format to the input instruction format if the output block format does not assert an access status of the source register.

6. The method of claim 4 wherein updating comprises:

setting the output block format to the input instruction format if the output block format is different from the input instruction format and the output block format asserts an access status of the source register.

7. The method of claim 4 wherein updating comprises:

setting the output block format to the output instruction format for the source register being used as output of the source instruction.

8. The method of claim 1 further comprising:

emitting a target instruction sequence corresponding to the source instruction into the target block of code;

emitting a block inconsistency check code into prefix of the target block of code; and

emitting a format update code to update a format register associated register format into the suffix of the target block of code.

9. The method of claim 1 wherein determining the register format comprises:

determining one of an access status, a packed single precision format, a packed double precision format, and a packed integer format.

10. The method of claim 8 wherein emitting the block inconsistency check code comprises:

emitting the block inconsistency check code to be executed during an execution phase following the translation phase.

11. A computer program product comprising:

a machine useable medium having program code embedded therein, the program code comprising:

computer readable program code to determine a register format of a source register operated on by a source instruction in a source block of code, the register format including an input instruction format and an output block format of the source block of code, the source block of code running in a source architecture, the source register having multiple formats and being used as an input of the source instruction, the input instruction format containing format of the source register expected by the source instruction, the output block format containing format of the source register after the source block of code is executed; and

computer readable program code to detect an instruction format inconsistency between the source register and a target register of a target architecture during a translation phase of a binary translation that translates the source block of code into a target block of code running in the target architecture.

12. The computer program product of claim 11 wherein the computer readable program code to detect the instruction format inconsistency comprises:

computer readable program code to compare the output block format to the input instruction format if the output block format asserts an access status of the source register.

13. The computer program product of claim 12 further comprising:

computer readable program code to emit a conversion code to convert the source register from the output block format to the input instruction format into the target block of code during the translation phase if the output block format is different from the input instruction format and the output block format asserts an access status of the source register.

14. The computer program product of claim 11 further comprising:

computer readable program code to update an input block format and the output block format, the input block format containing format of the source register expected by the source block of code before execution.

15. The computer program product of claim 14 wherein the computer readable program code to update comprises:

computer readable program code to set the input block format and the output block format to the input instruction format if the output block format does not assert an access status of the source register.

16. The computer program product of claim 14 wherein the computer readable program code to update comprises:

computer readable program code to set the output block format to the input instruction format if the output block format is different from the input instruction format and the output block format asserts an access status of the source register.

17. The computer program product of claim 14 wherein the computer readable program code to update comprises:

computer readable program code to set the output block format to the output instruction format for the source register being used as output of the source instruction.

18. The computer program product of claim 11 further comprising:

computer readable program code to emit a target instruction sequence corresponding to the source instruction into the target block of code;

computer readable program code to emit a block inconsistency check code into prefix of the target block of code; and

computer readable program code to emit a format update code to update a format register associated register format into the suffix of the target block of code.

19. The computer program product of claim 11 wherein the computer readable program code to determine the register format comprises:

computer readable program code to determine one of an access status, a packed single precision format, a packed double precision format, and a packed integer format.

20. The computer program product of claim 18 wherein the computer readable program code to emit the block inconsistency check code comprises:

computer readable program code to emit the block inconsistency check code to be executed during an execution phase following the translation phase.

21. A system comprising:

a processor; and

a memory coupled to the processor to store program code, the program code, when executed, causing the processor to:

determine a register format of a source register operated on by a source instruction in a source block of code, the register format including an input instruction format and an output block format of the source block of code, the source block of code running in a source architecture, the source register having multiple formats and being used as an input of the source instruction, the input instruction format containing format of the source register expected by the source instruction, the output block format containing format of the source register after the source block of code is executed; and

detect an instruction format inconsistency between the source register and a target register of a target architecture during a translation phase of a binary translation that translates the source block of code into a target block of code running in the target architecture.

22. The system of claim 21 wherein the program code causing the processor to detect the instruction format inconsistency causes the processor to:

compare the output block format to the input instruction format if the output block format asserts an access status of the source register.

23. The system of claim 22 wherein the program code further causing the processor to:

emit a conversion code to convert the source register from the output block format to the input instruction format into the target block of code during the translation phase if the output block format is different from the input instruction format and the output block format asserts an access status of the source register.

24. The system of claim 21 wherein the program code further causing the processor to:

update an input block format and the output block format, the input block format containing format of the source register expected by the source block of code before execution.

25. The system of claim 24 wherein the program code causing the processor to update causes the processor to:

set the input block format and the output block format to the input instruction format if the output block format does not assert an access status of the source register.

26. The system of claim 24 wherein the program code causing the processor to update causes the processor to:

set the output block format to the input instruction format if the output block format is different from the input instruction format and the output block format asserts an access status of the source register.

27. The system of claim 24 wherein the program code causing the processor to update causes the processor to:

set the output block format to the output instruction format for the source register being used as output of the source instruction.

28. The system of claim 21 the program code further causing the processor to:

emit a target instruction sequence corresponding to the source instruction into the target block of code;

emit a block inconsistency check code into prefix of the target block of code; and

emit a format update code to update a format register associated register format into the suffix of the target block of code.

29. The system of claim 21 wherein the program code causing the processor to determine the register format causes the processor to:

determine one of an access status, a packed single precision format, a packed double precision format, and a packed integer format.

30. The system of claim 28 wherein the program code causing the processor to emit the block inconsistency check code causes the processor to:

emit the block inconsistency check code to be executed during an execution phase following the translation phase.

31. A method comprising:

determining a register format of a source register operated on by a source instruction in a source block of code, the register format including an input block format and an output block format of the source block of code, the source block of code running in a source architecture, the source register having multiple formats and a format register associated with the register format, the input block format containing format of the source register expected by the source block of code, the output block format containing format of the source register after the source block of code is executed; and

detecting a block format inconsistency between the source register and a target register of a target architecture during an execution phase of a binary translation that translates the source block of code into a target block of code running in the target architecture.

32. The method of claim 31 wherein detecting the block format inconsistency comprises:

masking the format register with an input block format mask; and

comparing the masked format register with the input block format.

33. The method of claim 31 further comprising:

updating the format register upon exit of the target block of code.

34. The method of claim 33 wherein updating the format register comprises:

generating a first comparison result between the format register and the output block format;

masking the first comparison result by an output block format mask; and

generating a second comparison result between the format register and the masked first comparison result, the second comparison result corresponding to the updated format register.

35. The method of claim 32 further comprising:

executing a self-correcting code if the masked input block format is different than the input block format.

36. The method of claim 35 wherein executing comprises:

asserting a correction condition based on the format register and the input block format.

37. The method of claim 36 wherein asserting comprises:

comparing the format register to the input block format if the input block format asserts an access status of the source register.

38. The method of claim 36 further comprising:

converting the source register from format contained in the format register to format contained in the input block format; and

setting the format register to the input block format.

39. The method of claim 31 wherein determining the register format comprises:

determining one of an access status, a packed single precision format, a packed double precision format, and a packed integer format.

40. The method of claim 38 wherein detecting the block format inconsistency comprises:

detecting the block format inconsistency during the execution phase that follows a translation phase in the binary translation.

41. A computer program product comprising:

a machine useable medium having program code embedded therein, the program code comprising:

computer readable program code to determine a register format of a source register operated on by a source instruction in a source block of code, the register format including an input block format and an output block format of the source block of code, the source block of code running in a source architecture, the source register having multiple formats and a format register associated with the register format, the input block format containing format of the source register expected by the source block of code, the output block format containing format of the source register after the source block of code is executed; and

computer readable program code to detect a block format inconsistency between the source register and a target register of a target architecture during an execution phase of a binary translation that translates the source block of code into a target block of code running in the target architecture.

42. The computer program product of claim 41 wherein the computer readable program code to detect the block format inconsistency comprises:

computer readable program code to mask the format register with an input block format mask; and

computer readable program code to compare the masked format register with the input block format.

43. The computer program product of claim 41 further comprising:

computer readable program code to update the format register upon exit of the target block of code.

44. The computer program product of claim 43 wherein the computer readable program code to update the format register comprises:

computer readable program code to generate a first comparison result between the format register and the output block format;

computer readable program code to mask the first comparison result by an output block format mask; and

computer readable program code to generate a second comparison result between the format register and the masked first comparison result, the second comparison result corresponding to the updated format register.

45. The computer program product of claim 42 further comprising:

computer readable program code to execute a self-correcting code if the masked input block format is different than the input block format.

46. The computer program product of claim 45 wherein the computer readable program code to execute comprises:

computer readable program code to assert a correction condition based on the format register and the input block format.

47. The computer program product of claim 46 wherein the computer readable program code to assert comprises:

computer readable program code to compare the format register to the input block format if the input block format asserts an access status of the source register.

48. The computer program product of claim 46 further comprising:

computer readable program code to convert the source register from format contained in the format register to format contained in the input block format; and

computer readable program code to set the format register to the input block format.

49. The computer program product of claim 41 wherein the computer readable program code to determine the register format comprises:

computer readable program code to determine one of an access status, a packed single precision format, a packed double precision format, and a packed integer format.

50. The computer program product of claim 48 wherein the computer readable program code to detect the block format inconsistency comprises:

computer readable program code to detect the block format inconsistency during the execution phase that follows a translation phase in the binary translation.

51. A system comprising:

a processor; and

a memory coupled to the processor to store program code, the program code, when executed, causing the processor to:

determine a register format of a source register operated on by a source instruction in a source block of code, the register format including an input block format and an output block format of the source block of code, the source block of code running in a source architecture, the source register having multiple formats and a format register associated with the register format, the input block format containing format of the source register expected by the source block of code, the output block format containing format of the source register after the source block of code is executed; and

detect a block format inconsistency between the source register and a target register of a target architecture during an execution phase of a binary translation that translates the source block of code into a target block of code running in the target architecture.

52. The system of claim 51 wherein the program code causing the processor to detect the block format inconsistency causes the processor to:

mask the format register with an input block format mask; and

compare the masked format register with the input block format.

53. The system of claim 51 the program code further causing the processor to:

update the format register upon exit of the target block of code.

54. The system of claim 53 wherein the program code causing the processor to update the format register causes the processor to:

generate a first comparison result between the format register and the output block format;

mask the first comparison result by an output block format mask; and

generate a second comparison result between the format register and the masked first comparison result, the second comparison result corresponding to the updated format register.

55. The system of claim 52 wherein the program code further causing the processor to:

execute a self-correcting code if the masked input block format is different than the input block format.

56. The system of claim 55 the program code causing the processor to execute causes the processor to:

assert a correction condition based on the format register and the input block format.

57. The system of claim 56 the program code causing the processor to assert causes the processor to:

compare the format register to the input block format if the input block format asserts an access status of the source register.

58. The system of claim 56 wherein the program code further causing the processor to:

convert the source register from format contained in the format register to format contained in the input block format; and

set the format register to the input block format.

59. The system of claim 51 wherein the program code causing the processor to determine the register format causes the processor to:

determine one of an access status, a packed single precision format, a packed double precision format, and a packed integer format.

60. The system of claim 58 wherein the program code causing the processor to detect the block format inconsistency causes the processor to:

detect the block format inconsistency during the execution phase that follows a translation phase in the binary translation.